IN THE CLAIMS:

1. (Currently Amended) A method for automatically extracting a verification model from program source code independent of the source code, comprising the steps of:

generating a parse tree defining a control flow from the source code for procedural elements thereof;

identifying source code elements from the parse tree;

from the parse tree, generating text ~~source~~ strings for selected ones of the source code elements that correspond to function statements of the source code;

defining corresponding default conversions for translating the text ~~source~~ strings into a target language of a logic model checker; and

automatically generating in the target language a verification model ~~in the target language, wherein the verification model~~ that conforms to the control flow and to the corresponding default conversions for the selected ones of the source code elements.

2. (Currently Amended) The method of claim 1 comprising the further steps of:

optionally searching a conversion table for an entry associated with at least one of the text ~~source~~ strings, the entry including a translation for the at least one of the text ~~source~~ strings; and

substituting the translation for the corresponding default conversion for the at least one of the text ~~source~~ strings, wherein the verification model further conforms to the translation.

3. (Currently Amended) The method of claim 1 wherein the function ~~source code elements include basic~~ statements include elements of the source language selected from the group consisting of:

declarations,

assignments,

function calls,

return statements,

and boolean conditions ~~conditionals~~.

4. (Currently Amended) The method of claim 1 wherein the generating of ~~source~~ text strings includes the further step of expressing the ~~source~~ text strings in a canonical form.

5. (Original) The method of claim 1 wherein specifics of the corresponding default conversions can depend on a usage of the selected ones of the source code elements.

6. (Currently Amended) The method of claim 2 wherein the conversion table further includes samples of text ~~source~~ strings.

7. (Currently Amended) The method of claim 2 wherein the conversion table further includes classes of text ~~source~~ strings.

8. (Currently Amended) The method of claim 6 wherein the searching of the conversion table includes the step of pattern matching the at least one of the text ~~source~~ strings to the samples of text ~~source~~ strings.

9. (Currently Amended) The method of claim 7 wherein the searching of the conversion table includes the step of pattern matching the at least one of the text ~~source~~ strings to the classes of text ~~source~~ strings.

10. (Previously Presented) The method of claim 1 wherein the corresponding default conversions causes the translating of the text ~~source~~ strings to respective equivalent statements in the target language when the selected ones of the source code elements are fully relevant to a property to be tested, the translating of the text ~~source~~ strings to null statements in the target

language when the selected ones of the source code elements are irrelevant to the property to be tested, and the translating of the text ~~source~~ strings to preservation statements in the target language when the selected ones of the source code elements are partially relevant to the property to be tested, preservation statements being statements that preserve a relevant part of the text ~~source~~ strings and that suppress an irrelevant part of the text ~~source~~ strings.

11. (Currently Amended) The method of claim 2 where the generating a verification model step includes the further step of translating ones of the text ~~source~~ strings to a non-deterministic choice of possible outcomes.

12. (Currently Amended) The method of claim 2 wherein the generating a verification model step includes the step of populating the control flow with the translated text ~~source~~ strings.

13. (Original) The method of claim 1 wherein the default conversion includes a keep, the keep causing the generating of a verification model step to provide an equivalent statement in the target language.

14. (Previously Presented) The method of claim 1 wherein the default conversion comprises a hide, the hide causing the generating of a verification model step to provide a null statement in the target language.

15. (Currently Amended) The method of claim 1 herein the default conversion comprises a print, the print causing the generating of a verification model step to embed the respective text ~~source~~ strings in a print statement in the target language.

16. (Currently Amended) The method of claim 2 comprising the further step of simplifying the parse tree according to the translated text ~~source~~ strings.

17. (Previously Presented) The method of claim 16 wherein the simplifying step includes the steps of:

removing nodes corresponding to null statements;

removing nodes successive to false nodes; and

skipping selected nodes mapped to true.

18. (Currently Amended) The method of claim 3 comprising the further steps of:

collecting certain data object information for nodes in the parse tree corresponding to the function ~~basic~~ statements in the source code, the certain data object information including definition information and use information;

constructing a data dependency graph for the source code based upon the collected data object information, the data dependency graph having data dependency graph nodes corresponding to a data object, the data dependency graph having directed edges from first data dependency graph nodes to successive data dependency graph nodes if the successive data dependency graph nodes are used at least once in a definition of the first data dependency graph nodes;

determining a transitive closure for the data dependency graph dependency relation;

adding edges to the data dependency graph according to the transitive closure, the adding step providing a second data dependency graph;

for nodes corresponding to the function ~~basic~~ statements in the source code having translations other than hide or print, marking second data dependency graph data objects with identifiers corresponding to the definition information and the use information;

for nodes corresponding to the function ~~basic~~ statements in the source code having a hide translation; marking second data dependency graph data objects with a hide identifier; and

checking the second data dependency graph data objects for identifiers and the hide identifier.

19. (Currently Amended) A method for verifying that a software based system satisfies certain properties, the software based system having a source code, comprising the steps of:

automatically extracting a finite state model from the source code, the extracting step including the step of:

abstracting the source code statements based upon relevancies between the certain properties and the source code statements; and

expressing the finite state model in an input language for a logic model; and

checking the finite state model for the certain properties in the logic model checker.

20. (Currently Amended) A verification system for verifying that a source code system satisfies certain properties, the verification system ~~having a source code,~~ comprising:

a model extractor operable to automatically extract a finite state model from the source code, the model extractor implementing default conversions for translating selected source code elements and including:

a table of translation for translating other selected source code elements based upon defined abstractions, and

a translator responsive to the translations of the selected source code elements and the other selected source code elements for expressing the finite state model in an input language for a logic model checker, and

a logic model checker responsive to the certain properties and the finite state model for checking the finite state model for the certain properties.

21. (Currently Amended) The verification system of claim 20 wherein the model extractor further includes a parser for constructing a parse tree from the source code, wherein the translator translates selected text ~~source~~ strings generated from the parse tree.

22. (Currently Amended) The verification system of claim 21 wherein the model extractor further operates to provide a control flow from the parse tree and to populate the control flow with translated text ~~source~~ strings.

23. (Currently Amended) A method for automatically extracting a verification model from source code having a control flow for procedural elements of the source code, the method independent of the source code and comprising the steps of:

generating selected text ~~source~~ strings from the source code that represent the control flow;

translating ones of the selected text ~~source~~ strings to corresponding target language statements according to default conversions;

optionally searching a conversion table for user defined entries associated with the selected text ~~source~~ strings, the conversion table including a plurality of translations associated with various ones of the text ~~source~~ strings;

translating other ones of the selected text ~~source~~ strings to corresponding target language statements according to the user defined entries; and

automatically populating the control flow with the target language statements.